APPLICATION FOR UNITED STATES PATENT

in the name of

Lubomir D. Bourdev and Stephen N. Schiller

of

Adobe Systems Incorporated

for

Flattening Images

Fish & Richardson P.C.
2200 Sand Hill Road, Suite 100
Menlo Park, CA 94025
Tel.: (650) 322-5070
Fax: (650) 854-0875

ATTORNEY DOCKET:
07844-487001

DATE OF DEPOSIT: March 23, 2001

EXPRESS MAIL NO.: EL 572 620 696 US

FLATTENING IMAGES

BACKGROUND

The present invention relates to compositing and flattening images.

Many image processing computer programs, such as the Adobe Illustrator® program, available from Adobe Systems Incorporated of San Jose, California, build a final image by compositing two or more graphical objects together. The objects may be thought of as drawn on stacked sheets of acetate. An image thus composited can have multiple objects on multiple layers of acetate.

Each object typically includes data and compositing controls. An object can have a value that represents the object's color. An object can have a second value that represents the object's degree of transparency. This second value can cause the object to be represented as one that is opaque, translucent, or transparent. An object can be represented by an array of pixels or analytically, e.g., by using segment paths to represent shape outlines, or by other functions which map positions to data values. A shape outline of an object is the outline of the object's shape. For example, the shape outline of a dot is its circumference. Shape outlines will be referred to as outlines.

Some of the objects can overlap, some can be transparent, and some overlapping objects can be transparent. When a transparent object overlaps other objects, the underlying objects, i.e., those on an underlying sheet, are viewable.

Flattening is the process of converting an image containing transparency into a visually identical image that does not contain transparency. One method of flattening involves planar mapping, a process that divides overlapping objects into atomic regions based on the intersections of the overlapping objects' respective outlines. Where segment paths analytically represent outlines, this division is usually performed by calculating the intersections of the segment paths of the overlapping objects. Outlines used for planar mapping will be referred to as planar-mapping outlines.

When flattening involves text, calculating path intersections as described above can be computationally intensive and can significantly burden computing resources. Several factors contribute to this problem. First, text outlines are typically complex and, accordingly, require numerous segment paths for representation. An object having many segment paths usually requires more processor time and memory to flatten than does one having few

segment paths. Furthermore, text outlines are often curved and need to be described by second or third order expressions such as Bézier functions. Second or third order expressions usually require more processor time to flatten than do first order expressions. Additionally, because text outlines are often complex, the planar mapping of images involving text typically results in smaller and more numerous atomic regions than does planar mapping of images not involving text. An image having many small regions usually requires more processor time and memory to flatten than does an image having a few large regions. Thus, one problem in flattening images having text is that the process often requires great computing resources.

Another problem with flattening an image having text is that sometimes the font outlines required for planar mapping are unavailable. For example, some fonts have protected outlines and hence do not allow access to their segment paths. Additionally, some fonts such as bitmap fonts do not have outlines at all. Yet another problem with flattening an image having text arises from the conversion of native font information to outlines of text glyphs. Native font information, such as hinting data, is often not preserved in the conversion. Consequently, the quality of the flattened text can suffer, especially when the font size is small.

Objects that are not text can also present difficulties in flattening. For example, some objects have resolution-dependent outlines that are, thus, unavailable for planar mapping without a target resolution. These objects include ones having stroke instructions defining a width of the stroke based on a resolution of a selected output device. Also included are objects that have parametric bi-cubic or quadratic surfaces that define an outline based on a target resolution.

SUMMARY

The present invention provides methods, and apparatus including computer program products, for flattening an image.

Methods and apparatus in accordance with the invention efficiently employ computing resources to flatten an image. With respect to text, this efficiency is accomplished by defining a second outline for text objects, which usually have complex original outlines. The second outline encloses the text object and is simpler than the text object's original outline. The second outline requires few segment paths and no Bézier functions to represent.

The reduction in the number of segment paths and the absence of Bézier functions reduce the amount of computing resources needed to flatten an image. Moreover, the second outline reduces the number of regions formed from planar-mapping because the second outline is generally larger than original outline of text objects. The reduction in the number of regions reduces the amount of computing resources needed for flattening.

Methods and apparatus in accordance with the invention can also flatten images having objects with resolution-dependent outlines. This capability is accomplished by defining an outline for these types of objects. The outline defined in this case is similar to the second outline defined for text objects.

In general, in one aspect, the invention provides a method that includes receiving an image having multiple objects with each object having an original outline. Objects from among the multiple objects are selected to be abstracted objects. A planar-mapping outline is defined for each object. The planar-mapping outline of each selected object is an enclosing outline that encloses the object, and the planar-mapping outline of each object not selected is the original outline of the object. The image is divided into non-overlapping regions by using the planar-mapping outlines of all the objects. Each region is flattened.

In general, in another aspect, the invention provides a method that includes receiving an image having multiple objects with each object having an original outline. Objects from among the multiple objects are selected to be abstracted objects. A planar-mapping outline is defined for each object. The planar-mapping outline of each selected object is an enclosing outline that encloses the object, and the planar-mapping outline of each object not selected is the original outline of the object. The image is divided into regions that can overlap by using the planar-mapping outlines of the non-abstracted objects. None of the regions is intersected by a planar-mapping outline of a non-abstracted object. Each region is flattened.

The invention can be implemented to realize one or more of the following advantages. A system in accordance with the invention abstracts complex geometry out of planar maps and, thus, reduces the required computing resources such as memory and processor time. The system preserves native font information and hence improves the quality of fonts in a flattened image. Furthermore, the system can flatten images that have fonts for which segment paths are not available (such as bitmap fonts and protected fonts). The system can flatten images that have objects having resolution-dependent outlines. A method

implementing the invention can be applied not only to objects in PostScript® formats, but also to objects in other formats such as the Folio, TrueType, and IKARUS formats.

The details of one or more implementations of the invention are set forth in the accompanying drawings and the description below. Other features, objects, and advantages of the invention will be apparent from the description and drawings, and from the claims.

DESCRIPTION OF DRAWINGS

FIG. 1 is a flow chart of a method of flattening a layered image in accordance with the invention.

FIG. 2 shows an oriented bounding box of an object.

FIG. 3 is a flow chart of a method of flattening a region in accordance with the invention.

FIG. 4 is a flow chart of a method of determining background and foreground in accordance with the invention.

FIG. 5A shows a received image.

FIG. 5B shows outlines of objects in the received image.

FIG. 5C shows planar mapping the received image into regions.

FIG. 5D shows flattening a region of the received image.

FIG. 6A shows another received image.

FIG. 6B shows outlines of objects in the received image.

FIG. 6C shows planar mapping the received image into regions.

FIG. 6D shows flattening a region of the received image.

FIG. 7A shows an example of dividing an image into regions in an alternate implementation of the invention.

FIG. 7B shows an example of flattening a region in accordance with the alternate implementation of the invention.

FIG. 8 is a flow chart of a method of flattening in the alternate implementation of the invention.

Like reference symbols in the various drawings indicate like elements.

DETAILED DESCRIPTION

As shown in FIG. 1, a system that flattens an image having transparent objects performs a method 100 for flattening an image in accordance with the present invention. The

system receives an image having multiple objects (step 110). Some of the objects can be transparent so that underlying objects are viewable. Each object in the image has an original outline, i.e., the object's real outline. For example, if the image contains a circle and the letter A, the original outline of the circle is the circumference of the circle and the original outline of the letter A is the outline of the glyphs that represents the letter A.

Some of the objects can have original outlines that are unsuitable for use in planar mapping the received image. These unsuitable outlines include complex outlines, such as text outlines, and outlines that are resolution dependent, such as the above-described outlines that have resolution-dependent stroke instructions. For each object having an unsuitable outline, the system defines a second outline that is more suitable for use in planar mapping. Usually, this second outline is resolution-independent and is simpler than an object's original outline. Objects having the simpler outline will be referred to as abstracted objects. The planar-mapping outline for an abstracted object is its simpler outline.

For objects that have original outlines suitable for planar mapping, the system uses their original outlines. These objects will be referred to as non-abstracted objects. The planar-mapping outline for a non-abstracted object is its original outline.

In the present implementation, the system defines simpler outlines for text and for objects having resolution-dependent outlines. The system selects these objects to be abstracted objects (step 115) and defines a simpler outline as each object's planar-mapping outline (step 120). The simpler outline is a bounding box that encloses the object for which it is being defined. The bounding box is advantageously of a minimal size while still enclosing the selected object. Minimizing the size of the bounding box reduces the probability of the bounding box intersecting another object, consequently reducing the number of regions created in the planar mapping process. When necessary, the bounding box can be rotated or oriented with respect to its object to further minimize its size. For example, as shown in FIG. 2, oriented bounding box 210 is smaller than non-oriented bounding box 220. When there is a line of text, the system can treat the entire line as a single object and define a single bounding box that encloses the entire text line.

Optionally, the system can tailor the shape of the simpler outline for each selected object. Regardless of its shape, each outline still encloses the corresponding selected object. The shape of an outline depends partly on the object's shape and can be a rectangle, a triangle, a circle, an ellipse, or a polygon. For example, when an image has a substantially

circular selected object and a substantially rectangular selected object, the system defines a circular outline for the substantially circular selected object and a rectangular outline for the substantially rectangular selected object. Thus, in a given image, different objects can have simpler outlines of different shapes.

Referring back to FIG. 1, the system uses the original outlines of the non-abstracted objects to define their respective planar-mapping outlines (step 120). For example, a circle's planar-mapping outline is its original outline, i.e., its circumference.

The system uses the planar-mapping outlines of all objects to divide the received image into non-overlapping regions (step 130). The system performs this division by calculating the intersections of planar-mapping outlines. A planar-mapping process is further described in U.S. Patent Application No. 09/447,024, titled "Processing Complex Regions of Illustrated Artwork" and filed November 22, 1999, which is hereby incorporated by reference. Associated with each region are those abstracted and non-abstracted objects whose individual planar-mapping outline encloses the region.

The system flattens each region formed from planar mapping (step 140). FIG. 3 shows a flattening process 300, one method of flattening a region in accordance with the present invention. The system determines if there are any abstracted objects associated with the region (step 310). If there are none, then the system blends the colors of the non-abstracted objects (step 330). Blending includes determining a resulting color based on properties such as transparency, color value, and paint order of objects involved. A blending process is further described in the above-referenced U.S. Application No. 09/447,024. The system outputs the determined color value to an output stream of the method (step 330). The output stream of the method holds the data for outputting to a file or output device. If there are abstracted objects associated with the region, the system recursively determines background and foreground appearance for each abstracted object associated with the region (step 320).

As shown in FIG. 4, recursively determining background and foreground appearance includes selecting any abstracted object associated with the region (step 410). To determine the background appearance, the system starts a new flattening process 300, in the context of which the selected abstracted object is temporarily ignored (step 420). After completing the new flattening process and outputting the result (step 430), the system determines the foreground appearance by starting yet another flattening process 300, in the context of which

the currently selected abstracted object is temporarily treated as a non-abstracted object. Upon completing this most recently started flattening process, the system outputs the result clipped by the original outline of the currently selected abstracted object (step 450). Here, the system performs the clipping operation by setting the clipping path to the segment paths that represent the abstracted object's original outline. The recursion ends when all abstracted objects associated with the region have been selected.

FIGS. 5A-D show an example of flattening an image in accordance to method 100. As shown in FIG. 5A, the system receives an image 500 that contains transparent ellipses 510, 520 and abstracted object 530. The system uses original outlines 540 and 550, shown in FIG. 5B, to define planar-mapping outlines for ellipses 510 and 520, respectively (step 120). For abstracted object 530, the system defines its planar-mapping outline as bounding box 560 instead of original outline 570. From relatively simple planar-mapping outlines 540, 550, and 560, the system divides the received image into regions 571-77 as shown in FIG. 5C (step 130). The system then flattens regions 571-77 (step 140). For regions not having associated abstracted objects, such as regions 571-73, the system blends the colors of associated non-abstracted objects and outputs the determined color (step 330). For regions having associated abstracted objects, such as region 575, the system determines a background appearance by blending the colors of ellipses 510 and 520 (step 420), and then outputting the determined color (step 430). The system temporarily ignores abstracted object 530 during this operation. The system determines foreground appearance by blending the colors of ellipses 510, 520 and abstracted object 530 (step 440). The system then outputs the determined color clipped to the original outline of abstracted object 530 (step 450). The system represents each region by drawing the background and then drawing the abstracted object clipped to the region's outline. For example, as shown in FIG. 5D, the system draws background color 580 and then draws foreground color clipped to the region's outline 582 to obtain the regions final appearance 584. Note that the foreground color is clipped twice, once by the abstracted object's original outline and a second time by the region's outline. Flattening each region results in an image without transparency that is visually identical to the original image.

FIGS. 6A-D show another example of flattening an image in accordance to method 100. As shown in FIG. 6A, the received image 600 contains transparent ellipse 610, and overlapping transparent letters "A" 620 and "B" 630. The system defines

planar-mapping outlines (step 120). As shown in FIG. 6B, the system uses original outline 640 to define a planar-mapping outline for ellipse 610. For the letter "A" 620, the system defines its planar-mapping outline as bounding box 650 instead of original outline 660. Similarly, the system defines a planar-mapping outline for the letter "B" 630 as bounding box 670 instead of original outline 680. From relatively simple planar-mapping outlines 640, 650, and 670, the system divides the received image into regions 671-79 as shown in FIG. 6C (step 130). The system then flattens each of regions 671-79 (step 140). For regions having no more than one associated abstracted object, the flattening is the same as that described in FIG. 5D.

For regions having more than one associated abstracted object, such as region 671, the system recursively performs flattening process 300. As was shown in FIGS. 3 and 4, the system determines if there are any abstracted objects associated with region 671 (step 310). Since the letters "A" and "B" are both abstracted objects, the system determines background and foreground appearance for region 671 (step 320). The system selects the letter "B" (step 410) and starts a first recursion of flattening process 300 to determine the background of the region with respect to the letter "B", in the context of which the system temporarily ignores the currently selected abstracted object, the letter "B" (step 420). In the first recursion of flattening process 300, the system determines if there are any associated abstracted objects (step 310). Since there is one, i.e., the letter "A", the system determines background and foreground appearance (step 320). The system selects the remaining abstracted object (step 410), the letter "A" (as "B" is being ignored), and starts a second recursion of flattening process 300 to determine the background of the region with respect to the letter "A", in the context of which the system temporarily ignores the currently selected abstracted object, the letter "A" (step 420). In this second recursion, the system again determines if there are associated abstracted objects (step 310). There are none because both letters are being ignored. The system blends the colors of the remaining associated object, ellipse 610, and then outputs the determined color (step 330), thereby completing the second recursion of flattening process 300. This determined background is shown as element 690 of FIG. 6D. The system continues with the first recursion, determining the foreground of the region with respect to the letter "A" by starting yet another flattening process 300, a third recursion in the context of which the system treats the currently selected abstracted object, the letter "A", as a non-abstracted object and clips the determined color to the original outline of the letter "A"

(step 430). In the third recursion, the system determines if there are any associated abstracted objects (step 310). Since "B" is still being ignored and "A" is now set as a non-abstracted object, there are no associated abstracted objects and, accordingly, the system blends the colors of ellipse 630 and the letter "A" (step 330). The determined color, clipped by the original outline of the letter "A" is outputted (step 430), thereby completing the third and second recursions. At this point the system has determined the background of the region with respect to the letter "B" by determining the background and the foreground of the letter "A" (elements 690 and 692, respectively, of FIG. 6D).

The system now continues the first recursion, determining the foreground of the region with respect to the letter "B" by starting a fourth recursion, in the context of which the system temporarily treats the letter "B" as a non-abstracted object and clips the determined color to the original outline of the letter "B" (step 420). In the fourth recursion, the system determines if there are any associated abstracted objects (step 310). Since the system is treating the letter "B" as a non-abstracted object in this fourth recursion, the system selects the remaining associated abstracted object, the letter "A" (step 410), and starts a fifth recursion in the context of which the letter "A" is temporarily ignored (step 420). In this fifth recursion, the system determines if there are any associated abstracted objects (step 310). Since "B" is currently treated as a non-abstracted object and "A" is being ignored, there are no associated abstracted objects. Consequently, the system blends the colors of ellipse 710 and "B" (step 330), and outputs the determined color to the fourth recursion where the determined color is clipped by the original outline of the letter "B" (step 420), thereby completing the fifth recursion. This determined appearance is shown as element 694 in FIG. 6D. Continuing with the fourth recursion, the system starts a sixth recursion to determine the foreground of the region with respect to the letter "A", in the context of which the system temporarily treats the letter "A" as a non-abstracted object and clips the determined color to the original outline of the letter "A" (step 430). In this sixth recursion, the system currently treats both letters as non-abstracted objects. Hence, the system determines that there are no associated abstracted objects (step 310), and thus blends the colors of ellipse 610 and the letters "A" and "B" (step 330). The system outputs this determined color to the fourth recursion, clipped by the original outline of the letter "A" (step 450), thereby completing the sixth recursion. Note that in the current process of the fourth recursion, the outputted color is further clipped by the original outline of the letter "B" (step 450). Effectively, the

determined color is clipped by the intersection of the original outlines of the letters and outputted, thereby completing the fourth and first recursion and the underlying flattening process. This determined appearance is shown as element 696 in FIG. 6D. As shown in FIG. 6D, the system combines elements 690-96 of region 671 to obtain result 698.

In another implementation, dividing an image into regions can be performed so that abstracted objects do not divide outlines of other objects. In this implementation, a system uses the planar-mapping outlines of the non-abstracted objects, but not abstracted objects, to separate the image into regions that can overlap. Consequently, none of the regions formed are intersected by a planar-mapping outline of a non-abstracted object but can be intersected by a planar-mapping outline of an abstracted object.

For example, a system receives the image shown in FIG. 5A. The system defines planar-mapping outlines in a similar manner shown in FIG. 5B. Unlike the planar mapping shown in FIG. 5C, dividing the image into regions according to the alternate implementation, as shown in FIG. 7A, separates the image into regions 702-08, some of which overlap. As shown in FIG. 7B, the system represents each region having an associated abstracted object by drawing the abstracted object through the region. No drawing of the background is necessary when there are no abstracted objects or abstracted objects temporarily treated as non-abstracted objects because the background has not been clipped by these objects.

FIG. 8 shows an alternative to flattening process 320, the only difference being the addition of step 815. In flattening method 300, there are five possible types of objects: abstracted objects, non-abstracted objects, currently selected abstracted objects, currently ignored abstracted objects, and abstracted objects currently treated as non-abstracted objects. In step 815, the system determines whether there are abstracted objects, or abstracted objects currently treated as non-abstracted objects, excluding the currently-selected one. If there are, the system performs step 820. Otherwise, the system skips step 820 and proceeds to step 840.

The invention can be implemented in digital electronic circuitry, or in computer hardware, firmware, software, or in combinations of them. Apparatus of the invention can be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a programmable processor; and method steps of the invention can be performed by a programmable processor executing a program of instructions to perform functions of the invention by operating on input data and generating output. The

invention can be implemented advantageously in one or more computer programs that are executable on a programmable system including at least one programmable processor coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. Each computer program can be implemented in a high-level procedural or object-oriented programming language, or in assembly or machine language if desired; and in any case, the language can be a compiled or interpreted language. Suitable processors include, by way of example, both general and special purpose microprocessors. Generally, a processor will receive instructions and data from a read-only memory and/or a random access memory. Generally, a computer will include one or more mass storage devices for storing data files; such devices include magnetic disks, such as internal hard disks and removable disks; magneto-optical disks; and optical disks. Storage devices suitable for tangibly embodying computer program instructions and data include all forms of non-volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM disks. Any of the foregoing can be supplemented by, or incorporated in, ASICs (application-specific integrated circuits).

To provide for interaction with a user, the invention can be implemented on a computer system having a display device such as a monitor or LCD screen for displaying information to the user and a keyboard and a pointing device such as a mouse or a trackball by which the user can provide input to the computer system. The computer system can be programmed to provide a graphical user interface through which computer programs interact with users.

A number of implementations of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention.

For example, instead of a bounding box, the simpler outline can have any elementary shape such as an ellipse, a circle, a quadrangle, or a polygon. The polygon can have a pre-defined maximum number of edges. Alternatively, the system can rasterize a selected object and use the rasterized object's rectangular bounds as its planar-mapping outline.

When a region has multiple abstracted objects, instead of recursively determining background and foreground, the system iteratively determines background and foreground for the region.

Flattening a region (step 140) can involve rasterizing instead of blending and clipping. The system rasterizes portions of objects inside the region. The system then determines a color value for each pixel in the region based on the colors and other properties of objects represented by the pixel.

Method 100 can be applied to objects in non-PostScript® formats, such as objects in the Folio, TrueType, and IKARUS formats. Accordingly, other implementations are within the scope of the following claims.